View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by St Andrews Research Repository

Simplified Cloud Instance Selection

Chalee Boonprasop
School of Computer Science,
University of St Andrews, UK
cb330@st-andrews.ac.uk

Adam Barker
School of Computer Science,
University of St Andrews, UK
adam.barker@st-andrews.ac.uk

Abstract

Cloud computing delivers computational services to anyone over the internet. The cloud providers offer these services through a simplified billing model where customers can rent services based on the types of computing power they require. However, given the vast choice, it is complicated for a user to select the optimal instance types for a given workload or application. In this paper, we propose a user-friendly cloud instance recommendation system, which given a set of weighted coefficients representing the relevance of CPU, memory, storage and network along with a price, will recommend the best performing instances. The system only requires provider specified data about instance types and doesn't require costly cloud benchmarking. We evaluate our approach on Microsoft Azure across a number of different common workload types.

Keywords Cloud brokerage, Cloud computing

1. Introduction

The cloud computing market is one of the fastest growing sectors [1]. Many players in the cloud market want to offer the best deals possible, so the competition for the customer is high. Cloud providers create a lot of options for users to choose from in order to offer users with the most competitive prices and services. More choices are always good for the variety of products, but they also create a complicated market. There are several attempts at building a bridge between each provider [2]. The term "broker" is similar to the one used in the stock market. A cloud broker is a middleware between buyers and the goods in the market [3]. Users can ask the broker to buy, or in the cloud case rent a virtual machine (VM) instance.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CrossCloud'18, April 23–26, 2018, Porto, Portugal.
Copyright © 2018 ACM 978-1-4503-5653-4/18/04…$15.00.
http://dx.doi.org/10.1145/3195870.3195875

Most of the current cloud brokerage platforms do not offer recommendations or suggestions as to *which* provider and instance type are *most suitable* for a particular application. Moreover, the majority of cloud brokerages which do offer recommendations for instance selection utilize the cloud providers Service Level Agreements (SLA). As will all computers, the cloud instance performance can be derived from its specification. The difficulty of creating a cloud broker is that the performance of the cloud instance cannot be guaranteed. A lot of existing studies regarding the performance of cloud instances, [4] found that the cloud performance varies among multiple factors, such as time of the day, concurrence users, etc[5]. That being said, the variation in the performance of the same specification cloud instance is still small compared to the performance difference between two distinct instance type.

In this paper, we introduce a user-friendly simplified cloud instance selection based on weighted components. This model recommends users the best performing cloud instance according to a set of user preferences.

This paper makes the following research contributions:

- A user-friendly cloud recommender system, which only requires provider specified data about instances types.
- An objective function and algorithm which given a set of weighted coefficients representing the relevance of CPU, memory, storage and network along with a price will recommend the best performing instances.
- An evaluation our of approach on Microsoft Azure.

2. Related Work

The idea of cloud brokerage is not new. There are frameworks and services that deal with the cloud selection problem [8]. However, there has not been any real attempts to connect multiple providers with a suggestion system that aim to benefit the users.

The selection of cloud services is usually made manually, either by the users themselves or through an advisory company. The optimal deployment is not trivial due to the vast array of choices and configurations. Some frameworks are trying to help with the deployment problem. STRATOS [8] use normal topology as the resource acquisition decision. How-

ever, the optimization takes place with a constraint specified by the meta-data. There is no performance measurement involved in the selection process. The selection process can also be done based on the quality of service or through the dynamic negotiation of the service level agreements (SLA) [9]. Some of the cross-cloud platforms has been primarily developed to deal with the SLAs negotiation [10]. OPTIMIS is one of the toolkits that use past performance, maintenance, security, customer support, and legal aspect as a basis for the cloud selection. And recently, O'laughlin et al. [12] model the cloud brokerage into a market which users can buy the cloud instance with guaranteed performance. The performance data is from the historical benchmarking data.

Although, there are several attempts at dealing with the brokerage problem. The issue itself is that most of the data are not available to the researchers. Either the providers do not want to reveal much of their data due to security or maximizing their profit. Hence, it is necessary to build a system that uses only provided data.

3. Method of Selecting the Cloud Instances

In this paper, we introduce a components weighting method to aid the user in selecting the right cloud instance for a given application.

3.1 Performance Model

Performance weighting components have been applied in many research areas. In fact, it is the basic form of the objective function of a linear optimization problem. Moreover, the number of components such as CPU cores and the amount of RAM scale linearly in most setups in the cloud offering. Hence, a linear model is suitable for this kind of problem. The basic idea and the assumption being that there is a correlation between real-world cloud instance performance and the underlying low-level specification of the hardware which is specified on the cloud providers websites. Hence, we want to map the performance of the cloud instance to the right cloud workload using a simple linear model.

INSTANCE	CORE	RAM	TEMPORARY STORAGE	PAY AS YOU GO
B1S	1	1.00 GiB	2 GiB	$0.016/hour
B2S	2	4.00 GiB	8 GiB	$0.059/hour
B1MS	1	2.00 GiB	4 GiB	$0.029/hour

Figure 3.1 Typical cloud instance specification on a Provider's website [13].

Most of the computer workloads performance depends on three main components; CPU, Memory, and storage. For certain tasks, a strong CPU performance might benefit more than a high amount of Memory. Hence, the performance model consists of three main components, CPU, Memory, and storage. However, cloud computing by nature will require a fast network capability. Thus, network's performance is also added to the equation.

3.1.1 The Objective Function

Given that we are trying to choose the best performer instance from the sea of choices bounded to a constraint, it is logical to make the problem into a simple optimization problem. The objective function of our problem is represented as the weighted components performance number combine to a total performance of the cloud instance. It is given by the equation 1

$$w_c * perf_c + w_m * perf_m + w_s * perf_s + w_n * perf_n = perf_t \quad (1)$$

where, w_c, w_m, w_s, w_n are the weights of CPU, Memory, Storage and Network performance respectively. And, $perf_c$, $perf_m, perf_s, perf_n$, and $perf_t$ are the performance of CPU, Memory, Storage, Network and Total.

$perf_c, perf_m, perf_s, perf_n$ are normalized to make them have a similar impact on the objective function. The performance numbers are obtained directly from the provider website. In this paper, the $perf_c$ is the number of CPU cores. The $perf_m$ is the size of the memory. The $perf_s$ is the IO speed of the storage. Lastly, the $perf_n$ is the network speed of the cloud instance.

The weight is a four dimensional vector $w = [w_c, w_m, w_s, w_n]$, which is normalized to $w_c + w_m + w_s + w_n = 1$

Simply put the weights are the level of importance a user is attributing to each component for a given application, e.g., a more important component will get the value of 0.4 and the less important component will get the value of 0.1.

The only constraint is the budget of the user. Given the weights, we want to maximize the performance within the budget constraint.

3.1.2 Obtaining the Numbers

Benchmarking is a widely adopted method of obtaining the performance characteristics of a computer. It is also a reasonably accurate representation of the computing performance. However, there are too many parameters that affect the overall performance of computers. There is no universal benchmarking software that captures the total performance perfectly. Without the universal standard, getting a fair performance number for different configuration is non-trivial. Moreover, benchmarking process needs time to install and run, in many cases a lot of time.

Instead, it is faster to assume the performance numbers from the published specifications from the cloud providers are reputable sources and to utilize these data sets. Admittedly, these numbers do not reflect the performance of the cloud instance perfectly, but the trade-off in the speed is worth considering as a low-barrier way of driving a broker.

Assuming that our model above is accurate, the weight numbers need to be obtained. Typically, these weight num-

bers can be obtained from parameters optimization. The optimization can be done by running a lot of workload types on many of the cloud instances. Doing a parameter optimization is not practical for obvious reasons. Thus, the weight number can be set by the following two methods.

- User input: Provided that users understand the performance characteristics of their application.
- Pre-specified numbers: The weights can be set based on the description of the workloads.

The purpose of our work is to simplify the process of choosing a cloud instance. The weight numbers are provided using preliminary workload description.

3.2 Workload Classification

According to Singh and Chana [6], cloud workloads can be grouped into four main low-level measurable metrics. A CPU, memory, networking and storage workloads are the four distinct components that when adjusted can affect the workloads' performance. These four metrics will be used as a high-level interface that maps onto a cloud instance. For instance, performance-testing and computational science need high computing power. Hence, a higher CPU score is needed for this kind of workloads. The common cloud workloads are summarized below.

- Server Oriented: Websites, Technological computing, Endeavor software, Performance testing, On-line transaction processing, E-commerce, Central financial services, Storage and backup services.
- Client Oriented: Production applications, Software development and testing, Graphics oriented, Critical Internet applications.
- Mobile-Oriented: Mobile computing services.

These workload types are easily identifiable by the users. Hence if we can manage to map these workloads onto an easily measurable metrics, then the cloud instance selection will become more efficient.

3.3 Cloud Instance Data

In this paper, the cloud instance specifications are pulled directly from the Microsoft Azure service. The data includes all of the Ubuntu Linux general purposes and compute type instances. The data consisted of five parameters which are, the number of virtual compute units, the amount of RAM, the storage speed, and the network speed. The storage size is omitted as a result of premium storage space, where users can add more storage as needed. Additionally, the storage size does not directly affect the performance of the cloud instance unless it runs out.

3.4 Searching Algorithm

There are many methods of solving an optimization problem such as simplex algorithm and various type of metaheuristic algorithms [7]. However, for simplicity, the standard simulated annealing is used in this paper.

4. Experiments

In this section, the experiment is set up to see how well the model can capture the characteristic of deployed applications.

4.1 Dataset

Since there are so many instance types on Azure, we chose two representative price ranges to study. These two price ranges represent a lower end to mid-level cloud instances. The higher end of the spectrum is not considered because they are too few in numbers. So from the perspective of choosing, if your budget allows, there is no choice but to choose those offering.

The initial analysis is done on the cloud specification data itself, to see if there is any trend in the placing of the cloud instance price or not. The data are grouped using a mean-shift clustering algorithm since we do not know the number of clusters in the data. The result is that there are four main categories according to the feather of the cloud instance data. The first one is indicated as a normal instance. The second one is the high memory ratio instance. The third one is the high CPU ratio instance. And the last one is the high storage throughput instance. Which means that the providers are actively trying to assign the cloud specification to suit the user needs.

4.2 Testing

It is rather difficult to measure the total performance of the system without running real applications. Benchmarking software usually exercises specific parts of the system. However, we want a variety of workloads that exercise multiple parts of the system. Hence, only a handful of testing suites are suitable. A list of testing is chosen from openbenchmarking.org.

1. SQLite
2. Ebizzy
3. Scimark2

As mentioned before, the weight setting is rather abstract. Hence in this paper, the weight is set in the simplest way possible. The wight of a more important component will be set at 0.4 and the rest is set at 0.1.

The SQLite represents a database workload which should exercise heavily on the storage and memory performance. So, for the SQLite, the weights are assumed to be $w = [0.1, 0.4, 0.4, 0.1]$ The performance is indicated by the number of a second it takes to finish the task, lower the better.

The Ebizzy measures the amount of request that the webserver can handle during a certain amount of time. This test should represent a good mixture of network bandwidth and compute performance of the cloud instance. The weight for

the Ebizzy is set at $w = [0.4, 0.1, 0.1, 0.4]$. The result is the number of records that the server can process per second.

Lastly, the Scimark2 represents a compute-bounded workload. Scimark2 uses several scientific computing workloads such as matrix operations to measure the performance of the computer. These workloads usually require a lot of compute performance and high amount of memory bandwidth and capacity. The weight for the Scimark2 is set to be $w = [0.4, 0.4, 0.1, 0.1]$. In Scimark2, the score is measured in Mflops (Mega floating point operation per second) unit.

The three applications are deployed to each instance type. The result composed of three runs to measure the average values and the standard deviations.

It is obvious that the performance of the cloud is directly proportional to the amount of money spend. Hence the study is focusing on two price range of the Azure offering. The first budget point that tested is at 0.146$ per hour. This is the first quantile of the VM data. As shown in figure 4.2 below. The second budget that tested is at 0.398$ per hour. This price is the second quantile of the VM data. The test is done using only these two price ranges, mainly because of the density of cloud instance offering at these price range.

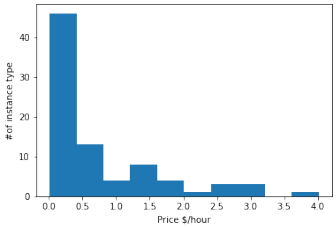

Figure 4.2: Distribution of the number of instance type offer by Azure and its price range.

4.3 Results

4.3.1 SQLite

Table 4.3.1: The ranking the cloud instances below 0.146$ per hour for the SQLite workload.

Ranking	Type	Cores	Mem	IOPs	Price
1	DS2 v2	2	7.0	6400	0.146
2	F2s	2	4.0	6400	0.1
3	E2v3	2	16.0	1000	0.133

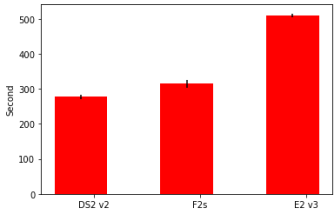

Figure 4.3.1 The time that each cloud instance takes to finish the SQLite workload (lower is better).

Table 4.3.1: The ranking and specification of the cloud instance below at the price 0.398$ per hour for the SQLite workload.

Ranking	Type	Cores	Mem	IOPs	Price
1	DS12 v2	4	28.0	12800	0.371
2	F8s v2	8	16.0	16000	0.338
3	D12v2	4	28.0	4000	0.371

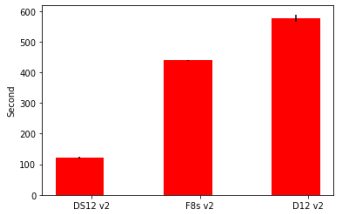

Figure 4.3.1 The time that each cloud instance takes to finish the SQLite workload (lower is better).

In the SQLite, the ranking is as expected. One interesting note is that F2s which is cheaper than E2 v3 is more powerful than E2 v3. Also in the case of mid-tier instances, some of them execute tasks slower than the cheaper instances.

4.3.2 Ebizzy

Table 4.3.2: The ranking and specification of the cloud instance below at the price 0.146$ per hour for the Ebizzy workload.

Ranking	Type	Cores	Mem	IOPs	Price
1	DS2 v2	2	7	6400	0.146
2	F2s	2	4	6400	0.1
3	F2 v2	2	4	4000	0.085

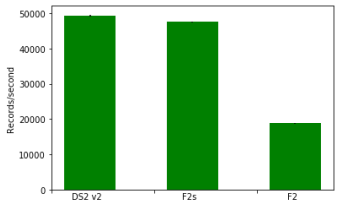

Figure 4.3.2 The amount of request per second that an instance can process (higher is better).

Table 4.3.2: The ranking and specification of the cloud instance below at the price 0.398$ per hour for the Ebizzy workload.

Ranking	Type	Cores	Mem	IOPs	Price
1	F8s v2	8	16	16000	0.338
2	DS12 v2	4	28	12800	0.371
2	D12v2	4	28.0	4000	0.371

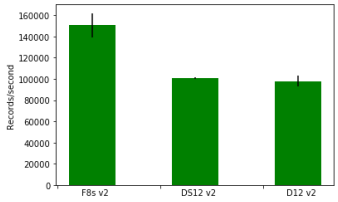

Figure 4.3.2 The amount of request per second that and instance can process (higher is better)

With web-server workloads, the model manages to produce an acceptable ranking.

4.3.3 Scimark2

Table 4.3.3: The ranking and specification of the cloud instance below at the price 0.146$ per hour for the Scimark2 workload.

Ranking	Type	Cores	Mem	IOPs	Price
1	A2mv2	2	16.0	2000	0.119
2	E2v3	2	16.0	1000	0.133
3	F2s	2	4	6400	0.1

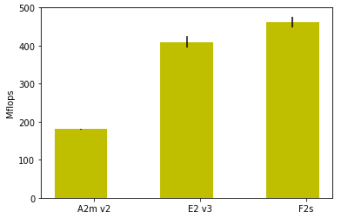

Figure 4.3.3 The MFLOP from Scimark2 workload(higher is better).

The result is completely opposite of what the model predicts.

Table 4.3.3: The ranking and specification of the cloud instance below at the price 0.398$ per hour for the Scimark2 workload.

Ranking	Type	Cores	Mem	IOPs	Price
1	B8MS	8	32	10800	0.339
2	DS12 v2	4	28	12800	0.371
3	F8s v2	8	16	16000	0.338

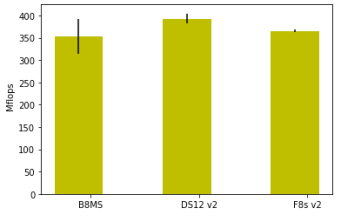

Figure 4.3.3 The MFLOP from Scimark2 workload(higher is better).

Similar to the cheaper instances, it is difficult to make use of the result. Additionally, the results show that an eight cores instance performs worst than a two cores instance. How is that possible?

To illustrate, all of the instances are tested again with the Cinebench. The Cinebench is a CPU heavy benchmarking software, that produces a score of both single and multi-core. The Scimark2 results show a strong correlation with the single core performance of the cloud instance.

Table 4.3.3: The CPU model of each cloud instance with respective Cinebench score.

Instance type	CPU model	Cinebench(single)
A2mv2	Xeon E5-2660	1.62
E2v3	Xeon E5-2673	1.80
F2s	Xeon E5-2673	1.83
B8MS	Xeon E5-2673	1.83
DS12 v2	Xeon E5-2673	1.83
F8s v2	Xeon Platinum 8168	1.9

4.4 Preliminary Workload Classification

Classification of data requires a targeted data. There type of data usually hard to find, especially sensitive data that is the workload type of customers. The next best approach is to analyses existing data such as server traces. Recently, Alibaba has provided researchers with a cluster trace. The trace consisted of 12 hours server load from 1300 machines running both online services and batch jobs. From running a mean-shift clustering algorithm, we found that the trace divided into six groups with various components load.

Table 4.4: centroids of components usage from Alibaba cluster trace.

CPU usage (%)	Memory usage (%)	Disk usage (%)
5.7	35	10
10	60	75
16	61	100
96	71	17
96	33	9.8
39	11	39

Table 4.4 shows that workloads can be divided into categories by usages of each components. Essentially, the weight numbers can be adjusted according to the table. However, we cannot take the result further than this without doing a more detail analysis. Nevertheless, there is a potential that our model can be utilized in solving the cloud selection problem.

5. Discussion

After the results are evaluated, all of the data is then statistically analyzed using Pearson correlation and linear regression analysis. Both SQLite and Ebizzy shows strong correlation and regression coefficient factor. However, Scimark2 does not share the same characteristics.

Using the correlation analysis on the result, Scimark2 does not have a strong correlation with any of the parameter that we have chosen. Which means that the model needs to be re-adjusted. Since in many cases, the number of CPU core is not the real indication of performance, due to many factors such as the CPU model, CPU frequency etc. The ranking is then recalculated using Cinebench CPU benchmark scores, instead of the number of cores originally used. The correlation coefficient does improve considerably, and the model is now able to report accurately. From the result, the Scimark2 does scale with a single core performance of the CPU. The result strongly suggests that in order for the model

to be more accurate, cloud workloads classification analysis is needed.

Regression analysis is used to identify the impact of each parameter. In our case, we would like to see how much of an impact each parameter has in each workload.

SQLite has a multiple R-value of 0.905 and has strong coefficient with both CPU and IO. This is in agreement with our initial assumption. EBizzy has a multiple R-value of 0.891 and has strong coefficient with CPU and IO speed and about half as strong on Memory. The result is similar to our initial assumption. Scimark2 has a multiple R-value of 0.26 and no significant impact with any of the parameter. The result is as expected since the correlation analysis also shows a similar trend. After changing the number of cores parameter to the cinebench score, the multiple R-value increases to 0.812.

Further observations are as follows. Some of the descriptions of the cloud instance specification are not accurate. Some of the instance, such as E2 v3 and D2 v3 is not correctly described. On the specification page, Azure lists them as two cores instance. However, both of them have one core company with a Hyperthread core. In some workloads, this parameter also needs to be reconsidered.

Workload classification couple with the weighted objective function can potentially be used as a cloud brokerage platform. As shown below the Pearson correlation indicate that both databased and web-application workloads has a high correlation coefficient with the pre-weighted objective function. The computational workload test, however, needs more investigations.

6. Conclusions

In this paper, we have demonstrated the feasibility of cloud brokerage based on a simple workload classification with a weighted objective function. The data used to drive the broker can be obtained from the provider easily, without conducting any timely and costly benchmarking. However, low-level data need further digging into such as CPU model, which in the case of Azure can be obtained from the description page; this will be more difficult with Amazon EC2 who give users an instance with random CPU model.

As a proof of concept, if the user can choose the desired group of their workloads, our model can reasonably suggest a suitable cloud instance for the user to deploy their work on.

With respect to future work, further research into the workload classification is required. As shown from statistical analysis, the result can be improved upon fine-tuning the weight using more data of the cloud workloads. The selection of parameters is also one area that could improve. Additionally, we would like to explore this technique across multiple cloud providers.

References

[1] Popovi, Kreimir, and eljko Hocenski. “Cloud computing security issues and challenges.” MIPRO, 2010 proceedings of the 33rd international convention. IEEE, 2010.

[2] Sundareswaran, Smitha, Anna Squicciarini, and Dan Lin. “A brokerage-based approach for cloud service selection.” Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on. IEEE, 2012.

[3] Elkhatib, Yehia. ”Mapping Cross-Cloud Systems: Challenges and Opportunities.” HotCloud. 2016.

[4] Chen, Yanpei, et al. “Towards understanding cloud performance tradeoffs using statistical workload analysis and replay.” University of California at Berkeley, Technical Report No. UCB/EECS-2010-81 (2010).

[5] Scheuner, Joel, et al. “Cloud workbench: Benchmarking iaas providers based on infrastructure-as-code.” Proceedings of the 24th International Conference on World Wide Web. ACM, 2015.

[6] Singh, Sukhpal, and Inderveer Chana. “A survey on resource scheduling in cloud computing: Issues and challenges.” Journal of grid computing 14.2 (2016): 217-264.

[7] Nelder, John A., and Roger Mead. “A simplex method for function minimization.” The computer journal 7.4 (1965): 308-313.

[8] Pawluk, Przemyslaw, et al. “Introducing STRATOS: A cloud broker service.” Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on. IEEE, 2012.

[9] Petcu, Dana. “Portability and interoperability between clouds: challenges and case study.” European Conference on a Service-Based Internet. Springer, Berlin, Heidelberg, 2011.

[10] Moscato, Francesco, et al. “An ontology for the cloud in mosaic.” (2011): 467-485.

[11] Ferrer, Ana Juan, et al. “OPTIMIS: A holistic approach to cloud service provisioning.” Future Generation Computer Systems 28.1 (2012): 66-77.

[12] OLoughlin, John, and Lee Gillam. “A performance brokerage for heterogeneous clouds.” Future Generation Computer Systems (2017).

[13] Microsoft, Azure. Windows Virtual Machines Pricing. Windows Virtual Machines — Microsoft Azure, 2017, azure.microsoft.com/en-gb/pricing/details/virtual-machines/windows/.